Domain Name System

The **Domain Name System** (**DNS**) is a hierarchical and distributed name service that provides a naming system for computers, services, and other resources on the Internet or other Internet Protocol (IP) networks. It associates various information with *domain names* (identification strings) assigned to each of the associated entities. Most prominently, it translates readily memorized domain names to the numerical IP addresses needed for locating and identifying computer services and devices with the underlying network protocols.[1] The Domain Name System has been an essential component of the functionality of the Internet since 1985.

The Domain Name System delegates the responsibility of assigning domain names and mapping those names to Internet resources by designating authoritative name servers for each domain. Network administrators may delegate authority over subdomains of their allocated name space to other name servers. This mechanism provides distributed and fault-tolerant service and was designed to avoid a single large central database. In addition, the DNS specifies the technical functionality of the database service that is at its core. It defines the DNS protocol, a detailed specification of the data structures and data communication exchanges used in the DNS, as part of the Internet protocol suite.

Domain Name Service Communication protocol	
Abbreviation	DNS
Purpose	To identify resources on networks
Developer(s)	Paul Mockapetris
Introduction	November 1983
OSI layer	Application layer
Port(s)	53
RFC(s)	RFC 1034, RFC 1035

Internet history timeline [show]

The Internet maintains two principal namespaces, the domain name hierarchy and the IP address spaces.[2] The Domain Name System maintains the domain name hierarchy and provides translation services between it and the address spaces. Internet name servers and a communication protocol implement the Domain Name System. A DNS name server is a server that stores the DNS records for a domain; a DNS name server responds with answers to queries against its database.

The most common types of records stored in the DNS database are for start of authority (SOA), IP addresses (A and AAAA), SMTP mail exchangers (MX), name servers (NS), pointers for reverse DNS lookups (PTR), and domain name aliases (CNAME). Although not intended to be a general-purpose database, DNS has been expanded over time to store records for other types of data for either automatic lookups, such as DNSSEC records, or for human queries such as

responsible person (RP) records. As a general-purpose database, the DNS has also been used in combating unsolicited email (spam) by storing blocklists. The DNS database is conventionally stored in a structured text file, the zone file, but other database systems are common.

The Domain Name System originally used the User Datagram Protocol (UDP) as transport over IP. Reliability, security, and privacy concerns spawned the use of the Transmission Control Protocol (TCP) as well as numerous other protocol developments.

Function

An often-used analogy to explain the DNS is that it serves as the phone book for the Internet by translating human-friendly computer hostnames into IP addresses. For example, the hostname `www.example.com` within the domain name example.com translates to the addresses *93.184.216.34* (IPv4) and *2606:2800:220:1:248:1893:25c8:1946* (IPv6). The DNS can be quickly and transparently updated, allowing a service's location on the network to change without affecting the end users, who continue to use the same hostname. Users take advantage of this when they use meaningful Uniform Resource Locators (URLs) and e-mail addresses without having to know how the computer actually locates the services.

An important and ubiquitous function of the DNS is its central role in distributed Internet services such as cloud services and content delivery networks.[3] When a user accesses a distributed Internet service using a URL, the domain name of the URL is translated to the IP address of a server that is proximal to the user. The key functionality of the DNS exploited here is that different users can *simultaneously* receive different translations for the *same* domain name, a key point of divergence from a traditional phone-book view of the DNS. This process of using the DNS to assign proximal servers to users is key to providing faster and more reliable responses on the Internet and is widely used by most major Internet services.[4]

The DNS reflects the structure of administrative responsibility on the Internet.[5] Each subdomain is a zone of administrative autonomy delegated to a manager. For zones operated by a registry, administrative information is often complemented by the registry's RDAP and WHOIS services. That data can be used to gain insight on, and track responsibility for, a given host on the Internet.[6]

History

Using a simpler, more memorable name in place of a host's numerical address dates back to the ARPANET era. The Stanford Research Institute (now SRI International) maintained a text file named HOSTS.TXT that mapped host names to the numerical addresses of computers on the ARPANET.[7][8] Elizabeth Feinler developed and maintained the first ARPANET directory.[9][10]

Maintenance of numerical addresses, called the Assigned Numbers List, was handled by Jon Postel at the University of Southern California's Information Sciences Institute (ISI), whose team worked closely with SRI.[11]

Addresses were assigned manually. Computers, including their hostnames and addresses, were added to the primary file by contacting the SRI Network Information Center (NIC), directed by Feinler, via telephone during business hours.[12] Later, Feinler set up a WHOIS directory on a server in the NIC for retrieval of information about resources, contacts, and entities.[13] She and her team developed the concept of domains.[13] Feinler suggested that domains should be based on the location of the physical address of the computer.[14] Computers at educational institutions would have the domain *edu*, for example.[15] She and her team managed the Host Naming Registry from 1972 to 1989.[16]

By the early 1980s, maintaining a single, centralized host table had become slow and unwieldy and the emerging network required an automated naming system to address technical and personnel issues. Postel directed the task of forging a compromise between five competing proposals of solutions to Paul Mockapetris. Mockapetris instead created the Domain Name System in 1983 while at the University of Southern California.[12][17]

The Internet Engineering Task Force published the original specifications in RFC 882 and RFC 883 in November 1983.[18][19] These were updated in RFC 973 in January 1986.[20]

In 1984, four UC Berkeley students, Douglas Terry, Mark Painter, David Riggle, and Songnian Zhou, wrote the first Unix name server implementation for the Berkeley Internet Name Domain, commonly referred to as BIND.[21] In 1985, Kevin Dunlap of DEC substantially revised the DNS implementation. Mike Karels, Phil Almquist, and Paul Vixie then took over BIND maintenance. Internet Systems Consortium was founded in 1994 by Rick Adams, Paul Vixie, and Carl Malamud, expressly to provide a home for BIND development and maintenance. BIND versions from 4.9.3 onward were developed and maintained by ISC, with support provided by ISC's sponsors. As co-architects/programmers, Bob Halley and Paul Vixie released the first production-ready version of BIND version 8 in May 1997. Since 2000, over 43 different core developers have worked on BIND.[22]

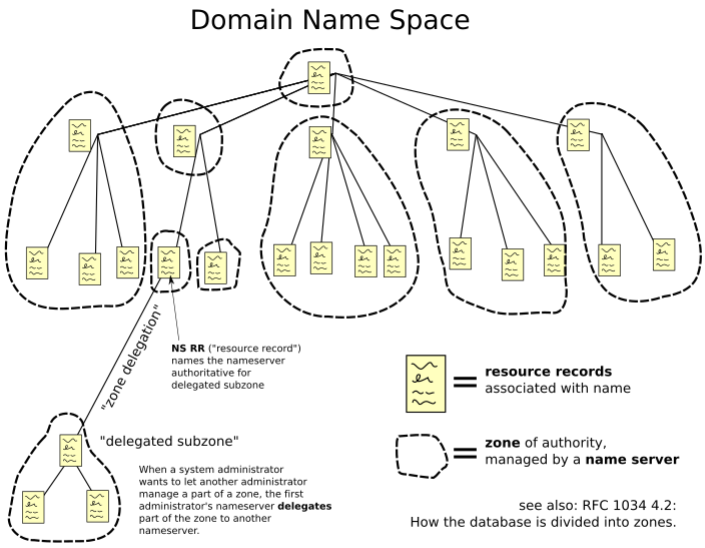
In November 1987, RFC 1034[23] and RFC 1035[5] superseded the 1983 DNS specifications. Several additional Request for Comments have proposed extensions to the core DNS protocols.[24]

Structure

Domain name space

The domain name space consists of a tree data structure. Each node or leaf in the tree has a *label* and zero or more *resource records* (RR), which hold information associated with the domain name. The domain name itself consists of the label, concatenated with the name of its parent node on the right, separated by a dot.[23]:§3.1

The tree sub-divides into *zones* beginning at the root zone. A DNS zone may consist of as many domains and subdomains as the zone manager chooses. DNS can also be partitioned according to *class* where the separate classes can be thought of as an array of parallel namespace trees.[23]:§4.2



The hierarchical Domain Name System for class *Internet*, organized into zones, each served by a name server

Administrative responsibility for any zone may be divided by creating additional zones. Authority over the new zone is said to be *delegated* to a designated name server. The parent zone ceases to be authoritative for the new zone.[23]:§4.2

Domain name syntax, internationalization

The definitive descriptions of the rules for forming domain names appear in RFC 1035, RFC 1123, RFC 2181, and RFC 5892. A domain name consists of one or more parts, technically called *labels*, that are conventionally concatenated, and delimited by dots, such as example.com.

The right-most label conveys the top-level domain; for example, the domain name www.example.com belongs to the top-level domain *com*.

The hierarchy of domains descends from right to left; each label to the left specifies a subdivision, or subdomain of the domain to the right. For example, the label *example* specifies a subdomain of the *com* domain, and *www* is a subdomain of example.com. This tree of subdivisions may have up to 127 levels.[25]

A label may contain zero to 63 characters, because the length is only allowed to take 6 bits. The null label of length zero is reserved for the root zone. The full domain name may not exceed the length of 253 characters in its textual representation (or 254 with the trailing dot).[23] In the internal binary representation of the DNS this maximum length of 253 requires 255 octets of storage, as it also stores the length of the first of many labels and adds last null byte.[5] 255 length is only achieved with at least 6 labels (counting the last null label).

Although no technical limitation exists to prevent domain name labels from using any character that is representable by an octet, hostnames use a preferred format and character set. The characters allowed in labels are a subset of the ASCII character set, consisting of characters *a* through *z*, *A* through *Z*, digits *0* through *9*, and hyphen. This rule is known as the *LDH rule* (letters, digits, hyphen). Domain names are interpreted in a case-independent manner.[26] Labels may not start or end with a hyphen.[27] An additional rule requires that top-level domain names should not be all-numeric.[27]

The limited set of ASCII characters permitted in the DNS prevented the representation of names and words of many languages in their native alphabets or scripts. To make this possible, ICANN approved the Internationalizing Domain Names in Applications (IDNA) system, by which user applications, such as web browsers, map Unicode strings into the valid DNS character set using Punycode. In 2009, ICANN approved the installation of internationalized domain name country code top-level domains (*ccTLD*s). In addition, many registries of the existing top-level domain names (*TLD*s) have adopted the IDNA system, guided by RFC 5890, RFC 5891, RFC 5892, RFC 5893.

Name servers

The Domain Name System is maintained by a distributed database system, which uses the client–server model. The nodes of this database are the name servers. Each domain has at least one authoritative DNS server that publishes information about that domain and the name servers of any domains subordinate to it. The top of the hierarchy is served by the root name servers, the servers to query when looking up (*resolving*) a TLD.

Authoritative name server

An *authoritative* name server is a name server that only gives answers to DNS queries from data that have been configured by an original source, for example, the domain administrator or by dynamic DNS methods, in contrast to answers obtained via a query to another name server that only maintains a cache of data.

An authoritative name server can either be a *primary* server or a *secondary* server. Historically the terms *master/slave* and *primary/secondary* were sometimes used interchangeably[28] but the current practice is to use the latter form. A primary server is a server that stores the original copies of all zone records. A secondary server uses a special automatic updating mechanism in the DNS protocol in communication with its primary to maintain an identical copy of the primary records.

Every DNS zone must be assigned a set of authoritative name servers. This set of servers is stored in the parent domain zone with name server (NS) records.

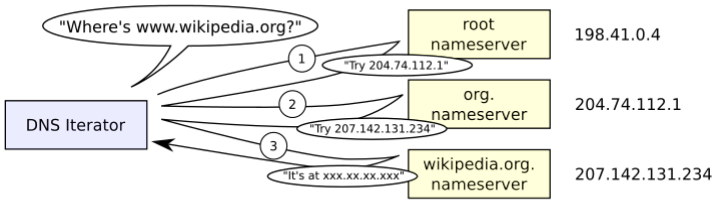
An authoritative server indicates its status of supplying definitive answers, deemed *authoritative*, by setting a protocol flag, called the "*Authoritative Answer*" (*AA*) bit in its responses.[5] This flag is usually reproduced prominently in the output of DNS administration query tools, such as dig, to indicate *that the responding name server is an authority for the domain name in question.*[5]

When a name server is designated as the authoritative server for a domain name for which it does not have authoritative data, it presents a type of error called a "lame delegation" or "lame response".[29][30]

Operation

Address resolution mechanism

Domain name resolvers determine the domain name servers responsible for the domain name in question by a sequence of queries starting with the right-most (top-level) domain label.



A DNS resolver that implements the iterative approach mandated by RFC 1034; in this case, the resolver consults three name servers to resolve the fully qualified domain name "www.wikipedia.org".

For proper operation of its domain name resolver, a network host is configured with an initial cache (*hints*) of the known addresses of the root name servers. The hints are updated periodically by an administrator by retrieving a dataset from a reliable source.

Assuming the resolver has no cached records to accelerate the process, the resolution process starts with a query to one of the root servers. In typical operation, the root servers do not answer directly, but respond with a referral to more authoritative servers, e.g., a query for "www.wikipedia.org" is referred to the *org* servers. The resolver now queries the servers referred to, and iteratively repeats this process until it receives an authoritative answer. The diagram illustrates this process for the host that is named by the fully qualified domain name "www.wikipedia.org".

This mechanism would place a large traffic burden on the root servers, if every resolution on the Internet required starting at the root. In practice caching is used in DNS servers to off-load the root servers, and as a result, root name servers actually are involved in only a relatively small fraction of all requests.

Recursive and caching name server

In theory, authoritative name servers are sufficient for the operation of the Internet. However, with only authoritative name servers operating, every DNS query must start with recursive queries at the root zone of the Domain Name System and each user system would have to implement resolver software capable of recursive operation.[31]

To improve efficiency, reduce DNS traffic across the Internet, and increase performance in end-user applications, the Domain Name System supports DNS cache servers which store DNS query results for a period of time determined in the configuration (*time-to-live*) of the domain name record in question. Typically, such caching DNS servers also implement the recursive algorithm necessary to resolve a given name starting with the DNS root through to the authoritative name servers of the queried domain. With this function implemented in the name server, user applications gain efficiency in design and operation.

The combination of DNS caching and recursive functions in a name server is not mandatory; the functions can be implemented independently in servers for special purposes.

Internet service providers typically provide recursive and caching name servers for their customers. In addition, many home networking routers implement DNS caches and recursion to improve efficiency in the local network.

DNS resolvers

The client side of the DNS is called a DNS resolver. A resolver is responsible for initiating and sequencing the queries that ultimately lead to a full resolution (translation) of the resource sought, e.g., translation of a domain name into an IP address. DNS resolvers are classified by a variety of query methods, such as *recursive*, *non-recursive*, and *iterative*. A resolution process may use a combination of these methods.[23]

In a *non-recursive query*, a DNS resolver queries a DNS server that provides a record either for which the server is authoritative, or it provides a partial result without querying other servers. In case of a caching DNS resolver, the non-recursive query of its local DNS cache delivers a result and reduces the load on upstream DNS servers by caching DNS resource records for a period of time after an initial response from upstream DNS servers.

In a *recursive query*, a DNS resolver queries a single DNS server, which may in turn query other DNS servers on behalf of the requester. For example, a simple stub resolver running on a home router typically makes a recursive query to the DNS server run by the user's ISP. A recursive query is one for which the DNS server answers the query completely by querying other name servers as needed. In typical operation, a client issues a recursive query to a caching recursive DNS server, which subsequently issues non-recursive queries to determine the answer and send a single answer back to the client. The resolver, or another DNS server acting recursively on behalf of the resolver, negotiates use of recursive service using bits in the query headers. DNS servers are not required to support recursive queries.

The *iterative query* procedure is a process in which a DNS resolver queries a chain of one or more DNS servers. Each server refers the client to the next server in the chain, until the current server can fully resolve the request. For example, a possible resolution of www.example.com would query a global root server, then a "com" server, and finally an "example.com" server.

Circular dependencies and glue records

Name servers in delegations are identified by name, rather than by IP address. This means that a resolving name server must issue another DNS request to find out the IP address of the server to which it has been referred. If the name given in the delegation is a subdomain of the domain for which the delegation is being provided, there is a circular dependency.

In this case, the name server providing the delegation must also provide one or more IP addresses for the authoritative name server mentioned in the delegation. This information is called *glue*. The delegating name server provides this glue in the form of records in the *additional*

section of the DNS response, and provides the delegation in the *authority section* of the response. A glue record is a combination of the name server and IP address.

For example, if the authoritative name server for example.org is ns1.example.org, a computer trying to resolve www.example.org first resolves ns1.example.org. As ns1 is contained in example.org, this requires resolving example.org first, which presents a circular dependency. To break the dependency, the name server for the top level domain org includes glue along with the delegation for example.org. The glue records are address records that provide IP addresses for ns1.example.org. The resolver uses one or more of these IP addresses to query one of the domain's authoritative servers, which allows it to complete the DNS query.

Record caching

A common approach to reduce the burden on DNS servers is to cache the results of name resolution locally or on intermediary resolver hosts. Each DNS query result comes with a time to live (TTL), which indicates how long the information remains valid before it needs to be discarded or refreshed. This TTL is determined by the administrator of the authoritative DNS server and can range from a few seconds to several days or even weeks.[32]

As a result of this distributed caching architecture, changes to DNS records do not propagate throughout the network immediately, but require all caches to expire and to be refreshed after the TTL. RFC 1912 conveys basic rules for determining appropriate TTL values.

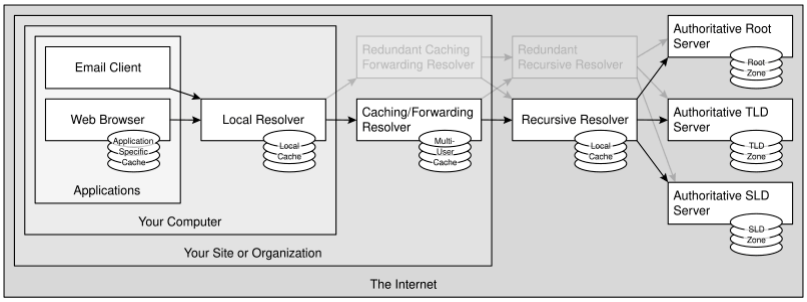
Some resolvers may override TTL values, as the protocol supports caching for up to sixty-eight years or no caching at all. Negative caching, i.e. the caching of the fact of non-existence of a record, is determined by name servers authoritative for a zone which must include the Start of Authority (SOA) record when reporting no data of the requested type exists. The value of the *minimum* field of the SOA record and the TTL of the SOA itself is used to establish the TTL for the negative answer.

Reverse lookup

A reverse DNS lookup is a query of the DNS for domain names when the IP address is known. Multiple domain names may be associated with an IP address. The DNS stores IP addresses in the form of domain names as specially formatted names in pointer (PTR) records within the infrastructure top-level domain arpa. For IPv4, the domain is in-addr.arpa. For IPv6, the reverse lookup domain is ip6.arpa. The IP address is represented as a name in reverse-ordered octet representation for IPv4, and reverse-ordered nibble representation for IPv6.

When performing a reverse lookup, the DNS client converts the address into these formats before querying the name for a PTR record following the delegation chain as for any DNS query. For example, assuming the IPv4 address 208.80.152.2 is assigned to Wikimedia, it is represented as a DNS name in reverse order: 2.152.80.208.in-addr.arpa. When the DNS resolver gets a pointer (PTR) request, it begins by querying the root servers, which point to the servers of American Registry for Internet Numbers (ARIN) for the 208.in-addr.arpa zone. ARIN's servers delegate 152.80.208.in-addr.arpa to Wikimedia to which the resolver sends another query for 2.152.80.208.in-addr.arpa, which results in an authoritative response.

Client lookup



DNS resolution sequence

Users generally do not communicate directly with a DNS resolver. Instead DNS resolution takes place transparently in applications such as web browsers, e-mail clients, and other Internet applications. When an application makes a request that requires a domain name lookup, such programs send a resolution request to the DNS resolver in the local operating system, which in turn handles the communications required.

The DNS resolver will almost invariably have a cache (see above) containing recent lookups. If the cache can provide the answer to the request, the resolver will return the value in the cache to the program that made the request. If the cache does not contain the answer, the resolver will send the request to one or more designated DNS servers. In the case of most home users, the Internet service provider to which the machine connects will usually supply this DNS server: such a user will either have configured that server's address manually or allowed DHCP to set it; however, where systems administrators have configured systems to use their own DNS servers, their DNS resolvers point to separately maintained name servers of the organization. In any event, the name server thus queried will follow the process outlined above, until it either successfully finds a result or does not. It then returns its results to the DNS resolver; assuming it has found a result, the resolver duly caches that result for future use, and hands the result back to the software which initiated the request.

Broken resolvers

Some large ISPs have configured their DNS servers to violate rules, such as by disobeying TTLs, or by indicating that a domain name does not exist just because one of its name servers does not respond.[33]

Some applications such as web browsers maintain an internal DNS cache to avoid repeated lookups via the network. This practice can add extra difficulty when debugging DNS issues as it obscures the history of such data. These caches typically use very short caching times on the order of one minute.[34]

Internet Explorer represents a notable exception: versions up to IE 3.x cache DNS records for 24 hours by default. Internet Explorer 4.x and later versions (up to IE 8) decrease the default timeout value to half an hour, which may be changed by modifying the default configuration.[35]

When Google Chrome detects issues with the DNS server it displays a specific error message.

Other applications

The Domain Name System includes several other functions and features.

Hostnames and IP addresses are not required to match in a one-to-one relationship. Multiple hostnames may correspond to a single IP address, which is useful in virtual hosting, in which many web sites are served from a single host. Alternatively, a single hostname may resolve to many IP addresses to facilitate fault tolerance and load distribution to multiple server instances across an enterprise or the global Internet.

DNS serves other purposes in addition to translating names to IP addresses. For instance, mail transfer agents use DNS to find the best mail server to deliver e-mail: An MX record provides a mapping between a domain and a mail exchanger; this can provide an additional layer of fault tolerance and load distribution.

The DNS is used for efficient storage and distribution of IP addresses of block-listed email hosts. A common method is to place the IP address of the subject host into the sub-domain of a higher level domain name, and to resolve that name to a record that indicates a positive or a negative indication.

For example:

- The address *203.0.113.5* is block-listed. It points to `5.113.0.203.blocklist.example`, which resolves to *127.0.0.1*.

- The address *203.0.113.6* is not block-listed and points to `6.113.0.203.blocklist.example`. This hostname is either not configured, or resolves to *127.0.0.2*.

E-mail servers can query blocklist.example to find out if a specific host connecting to them is in the block list. Many such block lists, either subscription-based or free of cost, are available for use by email administrators and anti-spam software.

To provide resilience in the event of computer or network failure, multiple DNS servers are usually provided for coverage of each domain. At the top level of global DNS, thirteen groups of root name servers exist, with additional "copies" of them distributed worldwide via anycast addressing.

Dynamic DNS (DDNS) updates a DNS server with a client IP address on-the-fly, for example, when moving between ISPs or mobile hot spots, or when the IP address changes administratively.

DNS message format

The DNS protocol uses two types of DNS messages, queries and responses; both have the same format. Each message consists of a header and four sections: question, answer, authority, and an additional space. A header field (*flags*) controls the content of these four sections.[23]

The header section consists of the following fields: *Identification*, *Flags*, *Number of questions*, *Number of answers*, *Number of authority resource records* (RRs), and *Number of additional RRs*. Each field is 16 bits long, and appears in the order given. The identification field is used to match responses with queries. After the flags word, the header ends with four 16-bit integers which contain the number of records in each of the sections that follow, in the same order.

DNS Header

Offset	Octet	0								1								2								3						
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	2
0	0	*Transaction ID*																*QR*	*OPCODE*				*AA*	*TC*	*RD*	*RA*	*Z*	*AD*	*CD*	*R*		
4	32	*Number of Questions*																*Number of Answers*													
8	64	*Number of Authority RRs*																*Number of additional RRs*													

Transaction ID: 16 bits

Transaction ID

Flags: 16 bits

The flag field consists of sub-fields as follows:

QR: 1 bit

Indicates if the message is a query (0) or a reply (1).

OPCODE: 4 bits

The type can be QUERY (standard query, 0), IQUERY (inverse query, 1), or STATUS (server status request, 2).

AA: 1 bit

Authoritative Answer, in a response, indicates if the DNS server is authoritative for the queried hostname.

TC: 1 bit

TrunCation, indicates that this message was truncated due to excessive length.

RD: 1 bit

Recursion Desired, indicates if the client means a recursive query.

RA: 1 bit

Recursion Available, in a response, indicates if the replying DNS server supports recursion.

Z: 1 bit; (Z) == 0

Zero, reserved for future use.

AD: 1 bit

Authentic Data, in a response, indicates if the replying DNS server verified the data.

CD: 1 bit

Checking Disabled, in a query, indicates that non-verified data is acceptable in a response.

RCODE: 4 bits

Response code, can be NOERROR (0), FORMERR (1, Format error), SERVFAIL (2), NXDOMAIN (3, Nonexistent domain), etc.[36]

Number of Questions: 16 bits

Number of Questions.

Number of Answers: 16 bits

Number of Answers.

Number of Authority RRs: 16 bits

Number of Authority Resource Records.

Number of Additional RRs: 16 bits

Number of Additional Resource Records.

Question section

The question section has a simpler format than the resource record format used in the other sections. Each question record (there is usually just one in the section) contains the following fields:

Resource record (RR) fields

Field	Description	Length (octets)
NAME	Name of the requested resource	Variable
TYPE	Type of RR (A, AAAA, MX, TXT, etc.)	2
CLASS	Class code	2

The domain name is broken into discrete labels which are concatenated; each label is prefixed by the length of that label.[37]

Resource records

The Domain Name System specifies a database of information elements for network resources. The types of information elements are categorized and organized with a list of DNS record types, the resource records (RRs). Each record has a type (name and number), an expiration time (time to live), a class, and type-specific data. Resource records of the same type are described as a *resource record set* (RRset), having no special ordering. DNS resolvers return the entire set upon query, but servers may implement round-robin ordering to achieve load balancing. In contrast, the Domain Name System Security Extensions (DNSSEC) work on the complete set of resource record in canonical order.

When sent over an Internet Protocol network, all records (answer, authority, and additional sections) use the common format specified in RFC 1035:[38]:§3

Resource record (RR) fields

Field	Description	Length (octets)
NAME	Name of the node to which this record pertains	Variable
TYPE	Type of RR in numeric form (e.g., 15 for MX RRs)	2
CLASS	Class code	2
TTL	Count of seconds that the RR stays valid (The maximum is $2^{31}-1$, which is about 68 years)	4
RDLENGTH	Length of RDATA field (specified in octets)	2
RDATA	Additional RR-specific data	Variable, as per RDLENGTH

NAME is the fully qualified domain name of the node in the tree. On the wire, the name may be shortened using label compression where ends of domain names mentioned earlier in the packet can be substituted for the end of the current domain name.

TYPE is the record type. It indicates the format of the data and it gives a hint of its intended use. For example, the *A* record is used to translate from a domain name to an IPv4 address, the *NS*

record lists which name servers can answer lookups on a DNS zone, and the *MX* record specifies the mail server used to handle mail for a domain specified in an e-mail address.

RDATA is data of type-specific relevance, such as the IP address for address records, or the priority and hostname for MX records. Well known record types may use label compression in the RDATA field, but "unknown" record types must not (RFC 3597).

The *CLASS* of a record is set to IN (for *Internet*) for common DNS records involving Internet hostnames, servers, or IP addresses. In addition, the classes Chaos (CH) and Hesiod (HS) exist.[38]:11 Each class is an independent name space with potentially different delegations of DNS zones.

In addition to resource records defined in a zone file, the domain name system also defines several request types that are used only in communication with other DNS nodes (*on the wire*), such as when performing zone transfers (AXFR/IXFR) or for EDNS (OPT).

Wildcard records

The domain name system supports wildcard DNS records which specify names that start with the *asterisk label*, `*`, e.g., `*.example`.[23][39] DNS records belonging to wildcard domain names specify rules for generating resource records within a single DNS zone by substituting whole labels with matching components of the query name, including any specified descendants. For example, in the following configuration, the DNS zone *x.example* specifies that all subdomains, including subdomains of subdomains, of *x.example* use the mail exchanger (MX) *a.x.example*. The AAAA record for *a.x.example* is needed to specify the mail exchanger IP address. As this has the result of excluding this domain name and its subdomains from the wildcard matches, an additional MX record for the subdomain *a.x.example*, as well as a wildcarded MX record for all of its subdomains, must also be defined in the DNS zone.

```
x.example.        MX    10 a.x.example.
*.x.example.      MX    10 a.x.example.
a.x.example.      MX    10 a.x.example.
*.a.x.example.    MX    10 a.x.example.
a.x.example.      AAAA  2001:db8::1
```

The role of wildcard records was refined in RFC 4592 (https://www.rfc-editor.org/rfc/rfc4592) , because the original definition in RFC 1034 (https://www.rfc-editor.org/rfc/rfc1034) was incomplete and resulted in misinterpretations by implementers.[39]

Protocol extensions

The original DNS protocol had limited provisions for extension with new features. In 1999, Paul Vixie published in RFC 2671 (superseded by RFC 6891) an extension mechanism, called Extension Mechanisms for DNS (EDNS) that introduced optional protocol elements without increasing overhead when not in use. This was accomplished through the OPT pseudo-resource record that only exists in wire transmissions of the protocol, but not in any zone files. Initial extensions were also suggested (EDNS0), such as increasing the DNS message size in UDP datagrams.

Dynamic zone updates

Dynamic DNS updates use the UPDATE DNS opcode to add or remove resource records dynamically from a zone database maintained on an authoritative DNS server.[40] This facility is useful to register network clients into the DNS when they boot or become otherwise available on the network. As a booting client may be assigned a different IP address each time from a DHCP server, it is not possible to provide static DNS assignments for such clients.

Transport protocols

From the time of its origin in 1983 the DNS has used the User Datagram Protocol (UDP) for transport over IP. Its limitations have motivated numerous protocol developments for reliability, security, privacy, and other criteria, in the following decades.

Conventional: DNS over UDP and TCP ports 53 (Do53)

1. UDP reserves port number 53 for servers listening to queries.[5] Such a query consists of a clear-text request sent in a single UDP packet from the client, responded to with a clear-text reply sent in a single UDP packet from the server. When the length of the answer exceeds 512 bytes and both client and server support Extension Mechanisms for DNS (EDNS), larger UDP packets may be used.[41] Use of DNS over UDP is limited by, among other things, its lack of transport-layer encryption, authentication, reliable delivery, and message length. In 1989, RFC 1123 specified optional Transmission Control Protocol (TCP) transport for DNS queries, replies and, particularly, zone transfers. Via fragmentation of long replies, TCP allows longer responses, reliable delivery, and re-use of long-lived connections between clients and servers. For larger responses, the server refers the client to TCP transport.

DNS over TLS (DoT)

DNS over TLS emerged as an IETF standard for encrypted DNS in 2016, utilizing Transport Layer Security (TLS) to protect the entire connection, rather than just the DNS payload. DoT servers listen on TCP port 853. RFC 7858 (https://www.rfc-editor.org/rfc/rfc7858) specifies that opportunistic encryption and authenticated encryption may be supported, but did not make either server or client authentication mandatory.

DNS over HTTPS (DoH)

DNS over HTTPS was developed as a competing standard for DNS query transport in 2018, tunneling DNS query data over HTTPS, which transports HTTP over TLS. DoH was promoted as a more web-friendly alternative to DNS since, like DNSCrypt, it uses TCP port 443, and thus looks similar to web traffic, though they are easily differentiable in practice without proper padding.[42]

DNS over QUIC (DoQ)

RFC 9250, published in 2022 by the Internet Engineering Task Force, describes DNS over QUIC. It has "privacy properties similar to DNS over TLS (DoT) [...], and latency characteristics similar to classic DNS over UDP". This method is not the same as DNS over HTTP/3.[43]

Oblivious DoH (ODoH) and predecessor Oblivious DNS (ODNS)

Oblivious DNS (ODNS) was invented and implemented by researchers at Princeton University and the University of Chicago as an extension to unencrypted DNS,[44] before DoH was standardized and widely deployed. Apple and Cloudflare subsequently deployed the technology in the context of DoH, as Oblivious DoH (ODoH).[45] ODoH combines ingress/egress separation (invented in ODNS) with DoH's HTTPS tunneling and TLS transport-layer encryption in a single protocol.[46]

DNS over Tor

DNS may be run over virtual private networks (VPNs) and tunneling protocols. The privacy gains of Oblivious DNS can be garnered through the use of the preexisting Tor network of ingress and egress nodes, paired with the transport-layer encryption provided by TLS.[47]

DNSCrypt

The DNSCrypt protocol, which was developed in 2011 outside the IETF standards framework, introduced DNS encryption on the downstream side of recursive resolvers, wherein clients encrypt query payloads using servers' public keys, which are published in the DNS (rather than relying upon third-party certificate authorities) and which may in turn be protected by DNSSEC signatures.[48] DNSCrypt uses either TCP port 443, the same port as HTTPS encrypted web traffic, or UDP port 443. This introduced not only privacy regarding the content of the query, but also a significant measure of firewall-traversal capability. In 2019, DNSCrypt was further extended to support an "anonymized" mode, similar to the proposed "Oblivious DNS", in which an ingress node receives a query which has been encrypted with the public key of a different server, and relays it to that server, which acts as an egress node, performing the recursive resolution.[49] Privacy of user/query pairs is created, since the ingress node does not know the content of the query, while the egress nodes does not know the identity of the client. DNSCrypt was first implemented in production by OpenDNS in December 2011. There are several free and open source software implementations that additionally integrate ODoH.[50] It is available for a variety of operating systems, including Unix, Apple iOS, Linux, Android, and Windows.

Security issues

Originally, security concerns were not major design considerations for DNS software or any software for deployment on the early Internet, as the network was not open for participation by the general public. However, the expansion of the Internet into the commercial sector in the 1990s changed the requirements for security measures to protect data integrity and user authentication.

Several vulnerability issues were discovered and exploited by malicious users. One such issue is DNS cache poisoning, in which data is distributed to caching resolvers under the pretense of being an authoritative origin server, thereby polluting the data store with potentially false information and long expiration times (time-to-live). Subsequently, legitimate application requests may be redirected to network hosts operated with malicious intent.

DNS responses traditionally do not have a cryptographic signature, leading to many attack possibilities; the Domain Name System Security Extensions (DNSSEC) modify DNS to add support for cryptographically signed responses.[51] DNSCurve has been proposed as an alternative to DNSSEC. Other extensions, such as TSIG, add support for cryptographic authentication between trusted peers and are commonly used to authorize zone transfer or dynamic update operations.

Techniques such as forward-confirmed reverse DNS can also be used to help validate DNS results.

DNS can also "leak" from otherwise secure or private connections, if attention is not paid to their configuration, and at times DNS has been used to bypass firewalls by malicious persons, and exfiltrate data, since it is often seen as innocuous.

DNS spoofing

Some domain names may be used to achieve spoofing effects. For example, `paypal.com` and `paypa1.com` are different names, yet users may be unable to distinguish them in a graphical user interface depending on the user's chosen typeface. In many fonts the letter *l* and the numeral *1* look very similar or even identical. This problem, known as the IDN homograph attack, is acute in systems that support internationalized domain names, as many character codes in ISO 10646 may appear identical on typical computer screens. This vulnerability is occasionally exploited in phishing.[52]

DNSMessenger

DNSMessenger[53][54][55][56] is a type of cyber attack technique that uses the DNS to communicate and control malware remotely without relying on conventional protocols that might raise red flags. The DNSMessenger attack is covert because DNS is primarily used for domain name resolution and is often not closely monitored by network security tools, making it an effective channel for attackers to exploit.

This technique involves the use of DNS TXT records to send commands to infected systems. Once malware has been surreptitiously installed on a victim's machine, it reaches out to a controlled domain to retrieve commands encoded in DNS text records. This form of malware communication is stealthy, as DNS requests are usually allowed through firewalls, and because DNS traffic is often seen as benign, these communications can bypass many network security defenses.

DNSMessenger attacks can enable a wide array of malicious activities, from data exfiltration to the delivery of additional payloads, all while remaining under the radar of traditional network security measures. Understanding and defending against such methods are crucial for maintaining robust cybersecurity.

Privacy and tracking issues

Originally designed as a public, hierarchical, distributed and heavily cached database, the DNS protocol has no confidentiality controls. User queries and nameserver responses are sent unencrypted, enabling network packet sniffing, DNS hijacking, DNS cache poisoning and man-in-the-middle attacks. This deficiency is commonly used by cybercriminals and network operators for marketing purposes, user authentication on captive portals and censorship.[57]

User privacy is further exposed by proposals for increasing the level of client IP information in DNS queries (RFC 7871) for the benefit of content delivery networks.

The main approaches that are in use to counter privacy issues with DNS include:

- VPNs, which move DNS resolution to the VPN operator and hide user traffic from the local ISP.
- Tor, which replaces traditional DNS resolution with anonymous .onion domains, hiding both name resolution and user traffic behind onion routing counter-surveillance.
- Proxies and public DNS servers, which move the actual DNS resolution to a trusted third-party provider.
    - Some public DNS servers may support security extensions such as DNS over HTTPS, DNS over TLS and DNSCrypt.

Solutions preventing DNS inspection by the local network operator have been criticized for thwarting corporate network security policies and Internet censorship. Public DNS servers are also criticized for contributing to the centralization of the Internet by placing control over DNS resolution in the hands of the few large companies which can afford to run public resolvers.[57]

> Google is the dominant provider of the platform in Android, the browser in Chrome, and the DNS resolver in the 8.8.8.8 service. Would this scenario be a case of a single corporate entity being in a position of overarching control of the entire namespace of the Internet? Netflix already fielded an app that used its own DNS resolution mechanism independent of the platform upon which the app was running. What if the Facebook app included DoH? What if Apple's iOS used a DoH-resolution mechanism to bypass local DNS resolution and steer all DNS queries from Apple's platforms to a set of Apple-operated name resolvers?
>
> —DNS Privacy and the IETF

Domain name registration

The right to use a domain name is delegated by domain name registrars which are accredited by the Internet Corporation for Assigned Names and Numbers (ICANN) or other organizations such as OpenNIC, that are charged with overseeing the name and number systems of the Internet. In addition to ICANN, each top-level domain (TLD) is maintained and serviced technically by an administrative organization, operating a registry. A *registry* is responsible for operating the database of names within its authoritative zone, although the term is most often used for TLDs. A *registrant* is a person or organization who asked for domain registration.[24] The registry receives registration information from each domain name *registrar*, which is authorized (accredited) to assign names in the corresponding zone and publishes the information using the WHOIS protocol. As of 2015, usage of RDAP is being considered.[58]

ICANN publishes the complete list of TLDs, TLD registries, and domain name registrars. Registrant information associated with domain names is maintained in an online database accessible with the WHOIS service. For most of the more than 290 country code top-level domains (ccTLDs), the domain registries maintain the WHOIS (Registrant, name servers, expiration dates, etc.) information. For instance, DENIC, Germany NIC, holds the DE domain data. From about 2001, most Generic top-level domain (gTLD) registries have adopted this so-called *thick* registry approach, i.e. keeping the WHOIS data in central registries instead of registrar databases.

For top-level domains on COM and NET, a *thin* registry model is used. The domain registry (e.g., GoDaddy, BigRock and PDR, VeriSign, etc., etc.) holds basic WHOIS data (i.e., registrar and name servers, etc.). Organizations, or registrants using ORG on the other hand, are on the Public Interest Registry exclusively.

Some domain name registries, often called *network information centers* (NIC), also function as registrars to end-users, in addition to providing access to the WHOIS datasets. The top-level domain registries, such as for the domains COM, NET, and ORG use a registry-registrar model consisting of many domain name registrars.[59] In this method of management, the registry only manages the domain name database and the relationship with the registrars. The *registrants* (users of a domain name) are customers of the registrar, in some cases through additional subcontracting of resellers.

See also

- Alternative DNS root
- Comparison of DNS server software

- Decentralized object location and routing
- Domain hijacking
- DNS hijacking
- DNS Long-Lived Queries
- DNS management software
- DNS over HTTPS
- DNS over TLS
- Hierarchical namespace
- IPv6 brokenness and DNS whitelisting
- Multicast DNS
- Public recursive name server
- resolv.conf
- Split-horizon DNS
- List of DNS record types
- List of managed DNS providers
- Zone file
- DNS leak

References

1. Wu, Hao; Dang, Xianglei; Wang, Lidong; He, Longtao (2016). "Information fusion-based method for distributed domain name system cache poisoning attack detection and identification" (https://onlinelibrary.wiley.com/doi/10.1049/iet-ifs.2014.0386) . *IET Information Security*. **10** (1): 37–44. doi:10.1049/iet-ifs.2014.0386 (https://doi.org/10.1049%2Fiet-ifs.2014.0386) . ISSN 1751-8717 (https://search.worldcat.org/issn/1751-8717) . S2CID 45091791 (https://api.semanticscholar.org/CorpusID:45091791) .
2. J. Postel, ed. (September 1981). *INTERNET PROTOCOL - DARPA INTERNET PROGRAM PROTOCOL SPECIFICATION* (https://datatracker.ietf.org/doc/html/rfc791) . IETF. doi:10.17487/RFC0791 (https://doi.org/10.17487%2FRFC0791) . STD 5. RFC 791 (https://datatracker.ietf.org/doc/html/rfc791) . IEN 128, 123, 111, 80, 54, 44, 41, 28, 26. *Internet Standard 5.* Obsoletes RFC 760 (https://www.rfc-editor.org/rfc/rfc760) . Updated by RFC 1349 (https://www.rfc-editor.org/rfc/rfc1349) , 2474 (https://www.rfc-editor.org/rfc/rfc2474) and 6864 (https://www.rfc-editor.org/rfc/rfc6864) .
3. J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Weihl. "Globally Distributed Content Delivery, IEEE Internet Computing, September/October 2002, pp. 50–58" (https://people.cs.umass.edu/~ramesh/Site/PUBLICATIONS_files/DMPPSW02.pdf) (PDF). Archived (https://web.archive.org/web/20150417121338/http://people.cs.umass.edu/~ramesh/Site/PUBLICATIONS_files/DMPPSW02.pdf) (PDF) from the original on 2015-04-17.

4. Nygren., E.; Sitaraman R. K.; Sun, J. (2010). "The Akamai Network: A Platform for High-Performance Internet Applications" (http://www.akamai.com/dl/technical_publications/network_overview_osr.pdf) (PDF). *ACM SIGOPS Operating Systems Review*. **44** (3): 2–19. doi:10.1145/1842733.1842736 (https://doi.org/10.1145%2F1842733.1842736) . S2CID 207181702 (https://api.semanticscholar.org/CorpusID:207181702) . Archived (https://web.archive.org/web/20101202153338/http://www.akamai.com/dl/technical_publications/network_overview_osr.pdf) (PDF) from the original on 2010-12-02. Retrieved November 19, 2012.

5. P. Mockapetris (November 1987). *DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION* (https://datatracker.ietf.org/doc/html/rfc1035) . Network Working Group. doi:10.17487/RFC1035 (https://doi.org/10.17487%2FRFC1035) . STD 13. RFC 1035 (https://datatracker.ietf.org/doc/html/rfc1035) . *Internet Standard 13*. Obsoletes RFC 882 (https://www.rfc-editor.org/rfc/rfc882) , 883 (https://www.rfc-editor.org/rfc/rfc883) and 973 (https://www.rfc-editor.org/rfc/rfc973) . Updated by RFC 1101 (https://www.rfc-editor.org/rfc/rfc1101) , 1183 (https://www.rfc-editor.org/rfc/rfc1183) , 1348 (https://www.rfc-editor.org/rfc/rfc1348) , 1876 (https://www.rfc-editor.org/rfc/rfc1876) , 1982 (https://www.rfc-editor.org/rfc/rfc1982) , 1995 (https://www.rfc-editor.org/rfc/rfc1995) , 1996 (https://www.rfc-editor.org/rfc/rfc1996) , 2065 (https://www.rfc-editor.org/rfc/rfc2065) , 2136 (https://www.rfc-editor.org/rfc/rfc2136) , 2137 (https://www.rfc-editor.org/rfc/rfc2137) , 2181 (https://www.rfc-editor.org/rfc/rfc2181) , 2308 (https://www.rfc-editor.org/rfc/rfc2308) , 2535 (https://www.rfc-editor.org/rfc/rfc2535) , 2673 (https://www.rfc-editor.org/rfc/rfc2673) , 2845 (https://www.rfc-editor.org/rfc/rfc2845) , 3425 (https://www.rfc-editor.org/rfc/rfc3425) , 3658 (https://www.rfc-editor.org/rfc/rfc3658) , 4033 (https://www.rfc-editor.org/rfc/rfc4033) , 4034 (https://www.rfc-editor.org/rfc/rfc4034) , 4035 (https://www.rfc-editor.org/rfc/rfc4035) , 4343 (https://www.rfc-editor.org/rfc/rfc4343) , 5936 (https://www.rfc-editor.org/rfc/rfc5936) , 5966 (https://www.rfc-editor.org/rfc/rfc5966) , 6604 (https://www.rfc-editor.org/rfc/rfc6604) , 7766 (https://www.rfc-editor.org/rfc/rfc7766) , 8482 (https://www.rfc-editor.org/rfc/rfc8482) , 8490 (https://www.rfc-editor.org/rfc/rfc8490) and 8767 (https://www.rfc-editor.org/rfc/rfc8767) .

6. Champika Wijayatunga (February 2015). "DNS Abuse Handling" (https://conference.apnic.net/data/39/dns-abuse-handling-final_1425362607.pdf) (PDF). APNIC. Archived (https://web.archive.org/web/20151222094305/https://conference.apnic.net/data/39/dns-abuse-handling-final_1425362607.pdf) (PDF) from the original on 2015-12-22. Retrieved 18 December 2016.

7. J. Klensin (February 2003). *Role of the Domain Name System (DNS)* (https://datatracker.ietf.org/doc/html/rfc3467) . Network Working Group. doi:10.17487/RFC3467 (https://doi.org/10.17487%2FRFC3467) . RFC 3467 (https://datatracker.ietf.org/doc/html/rfc3467) . *Informational*.

8. Liu, Cricket; Albitz, Paul (2006). *DNS and BIND* (5th ed.). O'Reilly Media. p. 3. ISBN 978-0-596-10057-5.
9. Evans 2018, p. 112.
10. Evans 2018, p. 113.
11. IEEE Annals [3B2-9] man2011030074.3d 29/7/011 11:54 Page 74
12. "Why Does the Net Still Work on Christmas? Paul Mockapetris - Internet Hall of Fame" (http://internethalloffame.org/blog/2012/07/23/why-does-net-still-work-christmas-paul-mockapetris) . *internethalloffame.org*. 23 July 2012.
13. Evans 2018, p. 119.
14. Evans 2018, p. 120.
15. Evans 2018, p. 120–121.
16. "Elizabeth Feinler" (https://web.archive.org/web/20180914182353/https://www.internethalloffame.org/inductees/elizabeth-feinler) . *Internet Hall of Fame*. Archived from the original (https://www.internethalloffame.org/inductees/elizabeth-feinler) on 14 September 2018. Retrieved 2018-11-25.
17. "Paul Mockapetris | Internet Hall of Fame" (https://internethalloffame.org/inductees/paul-mockapetris) . *internethalloffame.org*. Retrieved 2020-02-12.
18. Andrei Robachevsky (26 November 2013). "Happy 30th Birthday, DNS!" (https://www.internetsociety.org/blog/2013/11/happy-30th-birthday-dns) . Internet Society. Retrieved 18 December 2015.
19. Elizabeth Feinler, IEEE Annals, 3B2-9 man2011030074.3d 29/7/011 11:54 Page 74
20. P. Mockapetris (February 1986). *Domain System Changes and Observations* (https://datatracker.ietf.org/doc/html/rfc973) . Network Working Group. doi:10.17487/RFC0973 (https://doi.org/10.17487%2FRFC0973) . RFC 973 (https://datatracker.ietf.org/doc/html/rfc973) . *Obsolete.* Obsoleted by RFC 1034 (https://www.rfc-editor.org/rfc/rfc1034) and 1035 (https://www.rfc-editor.org/rfc/rfc1035) . Updates RFC 882 (https://www.rfc-editor.org/rfc/rfc882) and 883 (https://www.rfc-editor.org/rfc/rfc883) .
21. Terry, Douglas B.; et al. (June 12–15, 1984). "The Berkeley Internet Name Domain Server" (http://www.eecs.berkeley.edu/Pubs/TechRpts/1984/5957.html) . *Summer Conference, Salt Lake City 1984: Proceedings*. USENIX Association Software Tools Users Group. pp. 23–31.
22. Internet Systems Consortium. "The History of BIND" (https://www.isc.org/bindhistory/) . History of BIND. Archived (https://web.archive.org/web/20190630142752/https://www.isc.org/bindhistory/) from the original on 2019-06-30. Retrieved 4 April 2022.

23. P. Mockapetris (November 1987). *DOMAIN NAMES - CONCEPTS AND FACILITIES* (https://datatracker.ietf.org/doc/html/rfc1034) . Network Working Group. doi:10.17487/RFC1034 (https://doi.org/10.17487%2FRFC1034) . STD 13. RFC 1034 (https://datatracker.ietf.org/doc/html/rfc1034) . *Internet Standard 13.* Obsoletes RFC 882 (https://www.rfc-editor.org/rfc/rfc882) , 883 (https://www.rfc-editor.org/rfc/rfc883) and 973 (https://www.rfc-editor.org/rfc/rfc973) . Updated by RFC 1101 (https://www.rfc-editor.org/rfc/rfc1101) , 1183 (https://www.rfc-editor.org/rfc/rfc1183) , 1348 (https://www.rfc-editor.org/rfc/rfc1348) , 1876 (https://www.rfc-editor.org/rfc/rfc1876) , 1982 (https://www.rfc-editor.org/rfc/rfc1982) , 2065 (https://www.rfc-editor.org/rfc/rfc2065) , 2181 (https://www.rfc-editor.org/rfc/rfc2181) , 2308 (https://www.rfc-editor.org/rfc/rfc2308) , 2535 (https://www.rfc-editor.org/rfc/rfc2535) , 4033 (https://www.rfc-editor.org/rfc/rfc4033) , 4034 (https://www.rfc-editor.org/rfc/rfc4034) , 4035 (https://www.rfc-editor.org/rfc/rfc4035) , 4343 (https://www.rfc-editor.org/rfc/rfc4343) , 4592 (https://www.rfc-editor.org/rfc/rfc4592) , 5936 (https://www.rfc-editor.org/rfc/rfc5936) , 8020 (https://www.rfc-editor.org/rfc/rfc8020) , 8482 (https://www.rfc-editor.org/rfc/rfc8482) and 8767 (https://www.rfc-editor.org/rfc/rfc8767) .

24. P. Hoffman; A. Sullivan; K. Fujiwara (December 2015). *DNS Terminology* (https://datatracker.ietf.org/doc/html/rfc7719) . Internet Engineering Task Force. doi:10.17487/RFC7719 (https://doi.org/10.17487%2FRFC7719) . ISSN 2070-1721 (https://search.worldcat.org/issn/2070-1721) . RFC 7719 (https://datatracker.ietf.org/doc/html/rfc7719) . *Obsolete.* Obsoleted by RFC 8499 (https://www.rfc-editor.org/rfc/rfc8499) .

25. Lindsay, David (2007). *International Domain Name Law: ICANN and the UDRP*. Bloomsbury Publishing. p. 8. ISBN 978-1-84113-584-7.

26. D. Eastlake, 3rd (January 2006). *Domain Name System (DNS) Case Insensitivity Clarification* (https://datatracker.ietf.org/doc/html/rfc4343) . Network Working Group. doi:10.17487/RFC4343 (https://doi.org/10.17487%2FRFC4343) . RFC 4343 (https://datatracker.ietf.org/doc/html/rfc4343) . *Proposed Standard.* Updated by RFC 5890 (https://www.rfc-editor.org/rfc/rfc5890) . Updates RFC 1034 (https://www.rfc-editor.org/rfc/rfc1034) , 1035 (https://www.rfc-editor.org/rfc/rfc1035) and 2181 (https://www.rfc-editor.org/rfc/rfc2181) .

27. J. Klensin (February 2004). *Application Techniques for Checking and Transformation of Names* (https://datatracker.ietf.org/doc/html/rfc3696) . Network Working Group. doi:10.17487/RFC3696 (https://doi.org/10.17487%2FRFC3696) . RFC 3696 (https://datatracker.ietf.org/doc/html/rfc3696) . *Informational.*

28. Fujiwara, Kazunori; Sullivan, Andrew; Hoffman, Paul (2024). "DNS Terminology" (https://www.rfc-editor.org/rfc/rfc9499.html#section-6-4.42) . *tools.ietf.org*. doi:10.17487/RFC9499 (https://doi.org/10.17487%2FRFC9499) . Retrieved 2024-07-01.

29. Nemeth, Evi; Snyder, Garth; Hein, Trent R. (2006-10-30). *Linux Administration Handbook* (https://books.google.com/books?id=GB_O89fnz_sC&dq=%22lame+delegation%22&pg=PA475) . Addison-Wesley Professional. ISBN 978-0-13-700275-7.

30. Bissyande, Tegawendé F.; Sie, Oumarou (2017-10-09). *e-Infrastructure and e-Services for Developing Countries: 8th International Conference, AFRICOMM 2016, Ouagadougou, Burkina Faso, December 6-7, 2016, Proceedings* (https://books.google.com/books?id=YjE5DwAAQBAJ&dq=%22lame+delegation%22&pg=PA235) . Springer. ISBN 978-3-319-66742-3.

31. "DNS zone" (https://www.ionos.co.uk/digitalguide/server/know-how/dns-zone/) . *IONOS Digitalguide*. 27 January 2022. Retrieved 2022-03-31.

32. "What is DNS propagation?" (https://www.ionos.com/digitalguide/server/know-how/dns-propagation/) . *IONOS Digitalguide*. Retrieved 2022-04-22.

33. "Providers ignoring DNS TTL?" (http://ask.slashdot.org/story/05/04/18/198259/providers-ignoring-dns-ttl) . Slashdot. 2005. Retrieved 2012-04-07.

34. Ben Anderson (7 September 2011). "Ben Anderson: Why Web Browser DNS Caching Can Be A Bad Thing" (http://dyn.com/web-browser-dns-caching-bad-thing/) . Retrieved 20 October 2014.

35. "How Internet Explorer uses the cache for DNS host entries" (http://support.microsoft.com/default.aspx?scid=KB;en-us;263558) . Microsoft Corporation. 2004. Retrieved 2010-07-25.

36. "Domain Name System (DNS) Parameters" (https://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-6) . IANA. DNS RCODEs. Retrieved 14 June 2019.

37. James F. Kurose and Keith W. Ross, Computer Networking: A Top-Down Approach, 6th ed. Essex, England: Pearson Educ. Limited, 2012

38. D. Eastlake 3rd (April 2013). *Domain Name System (DNS) IANA Considerations* (https://datatracker.ietf.org/doc/html/rfc6895) . Internet Engineering Task Force. doi:10.17487/RFC6895 (https://doi.org/10.17487%2FRFC6895) . ISSN 2070-1721 (https://search.worldcat.org/issn/2070-1721) . BCP 42. RFC 6895 (https://datatracker.ietf.org/doc/html/rfc6895) . *Best Current Practice 42.* Obsoletes RFC 6195 (https://www.rfc-editor.org/rfc/rfc6195) . Updates RFC 2845 (https://www.rfc-editor.org/rfc/rfc2845) , 2930 (https://www.rfc-editor.org/rfc/rfc2930) , 1183 (https://www.rfc-editor.org/rfc/rfc1183) and 3597 (https://www.rfc-editor.org/rfc/rfc3597) .

39. E. Lewis (July 2006). *The Role of Wildcards in the Domain Name System* (https://datatracker.ietf.org/doc/html/rfc4592) . Network Working Group. doi:10.17487/RFC4592 (https://doi.org/10.17487%2FRFC4592) . RFC 4592 (https://datatracker.ietf.org/doc/html/rfc4592) . *Proposed Standard.* Updates RFC 2672 (https://www.rfc-editor.org/rfc/rfc2672) and 1034 (https://www.rfc-editor.org/rfc/rfc1034) .

40. S. Thomson; Y. Rekhter; J. Bound (April 1997). P. Vixie (ed.). *Dynamic Updates in the Domain Name System (DNS UPDATE)* (https://datatracker.ietf.org/doc/html/rfc2136) . IETF Network Working Group. doi:10.17487/RFC2136 (https://doi.org/10.17487%2FRFC2136) . RFC 2136 (https://datatracker.ietf.org/doc/html/rfc2136) . *Proposed Standard.* Updates RFC 1035 (https://www.rfc-editor.org/rfc/rfc1035) . Updated by RFC 3007 (https://www.rfc-editor.org/rfc/rfc3007) , 4033 (https://www.rfc-editor.org/rfc/rfc4033) , 4034 (https://www.rfc-editor.org/rfc/rfc4034) and 4035 (https://www.rfc-editor.org/rfc/rfc4035) .

41. RFC 2671 (https://www.rfc-editor.org/rfc/rfc2671) , *Extension Mechanisms for DNS (EDNS0)*, P. Vixie (August 1999)

42. Csikor, Levente; Divakaran, Dinil Mon (February 2021). "Privacy of DNS over HTTPS: Requiem for a Dream?" (https://raw.githubusercontent.com/cslev/doh_ml/main/DNS_over_HTTPS_identification.pdf) (PDF). National University of Singapore. "We investigate whether DoH traffic is distinguishable from encrypted Web traffic. To this end, we train a machine learning model to classify HTTPS traffic as either Web or DoH. With our DoH identification model in place, we show that an authoritarian ISP can identify ≈97.4% of the DoH packets correctly while only misclassifying 1 in 10,000 Web packets."

43. Huitema, Christian; Dickinson, Sara; Mankin, Allison (May 2022). *DNS over Dedicated QUIC Connections* (https://datatracker.ietf.org/doc/html/rfc9250) . Internet Engineering Task Force. doi:10.17487/RFC9250 (https://doi.org/10.17487%2FRFC9250) . RFC 9250 (https://datatracker.ietf.org/doc/html/rfc9250) .

44. Schmitt, Paul; Edmundson, Anne; Feamster, Nick (2019). "Oblivious DNS: Practical Privacy for DNS Queries" (https://petsymposium.org/2019/files/papers/issue2/popets-2019-0028.pdf) (PDF). *Privacy Enhancing Technologies*. **2019** (2): 228–244. arXiv:1806.00276 (https://arxiv.org/abs/1806.00276) . doi:10.2478/popets-2019-0028 (https://doi.org/10.2478%2Fpopets-2019-0028) . S2CID 44126163 (https://api.semanticscholar.org/CorpusID:44126163) . Archived (https://web.archive.org/web/20220121210624/https://petsymposium.org/2019/files/papers/issue2/popets-2019-0028.pdf) (PDF) from the original on 2022-01-21.

45. "Oblivious DNS Deployed by Cloudflare and Apple" (https://medium.com/noise-lab/oblivious-dns-deployed-by-cloudflare-and-apple-1522ccf53cab) . 9 December 2020. Retrieved 27 July 2022.

46. Pauly, Tommy (2 September 2021). "Oblivious DNS Over HTTPS" (https://datatracker.ietf.org/doc/draft-pauly-dprive-oblivious-doh/) . IETF.

47. Muffett, Alec (February 2021). " "No Port 53, Who Dis?" A Year of DNS over HTTPS over Tor" (https://www.ndss-symposium.org/wp-content/uploads/dnspriv21-03-paper.pdf) (PDF). Network and Distributed System Security Symposium. Archived (https://web.archive.org/web/20210321110839/https://www.ndss-symposium.org/wp-content/uploads/dnspriv21-03-paper.pdf) (PDF) from the original on 2021-03-21. "DNS over HTTPS (DoH) obviates many but not all of the risks, and its transport protocol (i.e. HTTPS) raises concerns of privacy due to (e.g.) 'cookies.' The Tor Network exists to provide TCP circuits with some freedom from tracking, surveillance, and blocking. Thus: In combination with Tor, DoH, and the principle of "Don't Do That, Then" (DDTT) to mitigate request fingerprinting, I describe DNS over HTTPS over Tor (DoHoT)."

48. Ulevitch, David (6 December 2011). "DNSCrypt – Critical, fundamental, and about time" (https://umbrella.cisco.com/blog/dnscrypt-critical-fundamental-and-about-time) . *Cisco Umbrella*. Archived (https://web.archive.org/web/20200701221715/https://umbrella.cisco.com/blog/dnscrypt-critical-fundamental-and-about-time) from the original on 1 July 2020.

49. "Anonymized DNSCrypt specification" (https://raw.githubusercontent.com/DNSCrypt/dnscrypt-protocol/master/ANONYMIZED-DNSCRYPT.txt) . *GitHub*. DNSCrypt. Archived (https://web.archive.org/web/20191025094649/https://raw.githubusercontent.com/DNSCrypt/dnscrypt-protocol/master/ANONYMIZED-DNSCRYPT.txt) from the original on 25 October 2019.

50. "Oblivious DoH · DNSCrypt/dnscrypt-proxy Wiki" (https://github.com/DNSCrypt/dnscrypt-proxy/wiki/Oblivious-DoH) . *GitHub*. DNSCrypt project. Retrieved 28 July 2022.

51. Herzberg, Amir; Shulman, Haya (2014-01-01). "Retrofitting Security into Network Protocols: The Case of DNSSEC" (https://ieeexplore.ieee.org/document/6756846) . *IEEE Internet Computing*. **18** (1): 66–71. doi:10.1109/MIC.2014.14 (https://doi.org/10.1109%2FMIC.2014.14) . ISSN 1089-7801 (https://search.worldcat.org/issn/1089-7801) . S2CID 12230888 (https://api.semanticscholar.org/CorpusID:12230888) .

52. APWG. "Global Phishing Survey: Domain Name Use and Trends in 1H2010." 10/15/2010 apwg.org (http://www.apwg.org/reports/APWG_GlobalPhishingSurvey_1H2010.pdf) Archived (https://web.archive.org/web/20121003212327/http://apwg.org/reports/APWG_GlobalPhishingSurvey_1H2010.pdf) 2012-10-03 at the Wayback Machine

53. "DNSMessenger (Malware Family)" (https://malpedia.caad.fkie.fraunhofer.de/details/win.dnsmessenger) . *malpedia.caad.fkie.fraunhofer.de*. Retrieved 2024-12-11.

54. Khandelwal, Swati (2017-03-06). "New Fileless Malware Uses DNS Queries To Receive PowerShell Commands" (https://thehackernews.com/2017/03/powershell-dns-malware.html) . *The Hacker News*. Retrieved 2024-12-11.

55. Brumaghin, Edmund (2017-03-02). "Covert Channels and Poor Decisions: The Tale of DNSMessenger" (https://blog.talosintelligence.com/dnsmessenger/) . *Cisco Talos Blog*. Retrieved 2024-12-11.

56. Bombal, David (2023-05-26). *It's DNS again 🤒 Did you know this Malware Hack?* (https://www.youtube.com/watch?v=slNe6z9gFv0) . Retrieved 2024-12-11 – via YouTube.

57. Huston, Geoff (July 2019). "DNS Privacy and the IETF" (http://ipj.dreamhosters.com/wp-content/uploads/2019/07/ipj222.pdf) (PDF). *The Internet Protocol Journal*. Archived (https://web.archive.org/web/20190930154208/http://ipj.dreamhosters.com/wp-content/uploads/2019/07/ipj222.pdf) (PDF) from the original on 2019-09-30.

58. "Registration Data Access Protocol (RDAP) Operational Profile for gTLD Registries and Registrars" (https://web.archive.org/web/20151222144443/https://www.icann.org/news/announcement-2015-12-03-en) . ICANN. 3 December 2015. Archived from the original (https://www.icann.org/news/announcement-2015-12-03-en) on 22 December 2015. Retrieved 18 December 2015.

59. "Find a Registrar" (http://www.verisign.com/en_US/domain-names/domain-registrar/index.xhtml) . VeriSign, Inc. Retrieved 18 December 2015.

Sources

- Evans, Claire L. (2018). *Broad Band: The Untold Story of the Women Who Made the Internet* (https://books.google.com/books?id=C8ouDwAAQBAJ&q=9780735211759&pg=PP1) . New York: Portfolio/Penguin. ISBN 9780735211759.

Further reading

Standards track

- RFC 1034 (https://www.rfc-editor.org/rfc/rfc1034) – "DOMAIN NAMES - CONCEPTS AND FACILITIES," *Internet Standard 13*.
- RFC 1035 (https://www.rfc-editor.org/rfc/rfc1035) – "DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION," *Internet Standard 13*.
- RFC 1123 (https://www.rfc-editor.org/rfc/rfc1123) – "Requirements for Internet Hosts -- Application and Support," *Internet Standard 3*.
- RFC 1995 (https://www.rfc-editor.org/rfc/rfc1995) – "Incremental Zone Transfer in DNS," *Proposed Standard*.
- RFC 1996 (https://www.rfc-editor.org/rfc/rfc1996) – "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)," *Proposed Standard*.

- RFC 2136 (https://www.rfc-editor.org/rfc/rfc2136) – "Dynamic Updates in the Domain Name System (DNS UPDATE)," *Proposed Standard*.
- RFC 2181 (https://www.rfc-editor.org/rfc/rfc2181) – "Clarifications to the DNS Specification," *Proposed Standard*.
- RFC 2308 (https://www.rfc-editor.org/rfc/rfc2308) – "Negative Caching of DNS Queries (DNS NCACHE)," *Proposed Standard*.
- RFC 3225 (https://www.rfc-editor.org/rfc/rfc3225) – "Indicating Resolver Support of DNSSEC," *Proposed Standard*.
- RFC 3226 (https://www.rfc-editor.org/rfc/rfc3226) – "DNSSEC and IPv6 A6 aware server/resolver message size requirements," *Proposed Standard*.
- RFC 3596 (https://www.rfc-editor.org/rfc/rfc3596) – "DNS Extensions to Support IP Version 6," *Internet Standard 88*.
- RFC 3597 (https://www.rfc-editor.org/rfc/rfc3597) – "Handling of Unknown DNS Resource Record (RR) Types," *Proposed Standard*.
- RFC 4343 (https://www.rfc-editor.org/rfc/rfc4343) – "Domain Name System (DNS) Case Insensitivity Clarification," *Proposed Standard*.
- RFC 4592 (https://www.rfc-editor.org/rfc/rfc4592) – "The Role of Wildcards in the Domain Name System," *Proposed Standard*.
- RFC 5001 (https://www.rfc-editor.org/rfc/rfc5001) – "DNS Name Server Identifier (NSID) Option," *Proposed Standard*.
- RFC 5011 (https://www.rfc-editor.org/rfc/rfc5011) – "Automated Updates of DNS Security (DNSSEC) Trust Anchors," *Internet Standard 74*.
- RFC 5452 (https://www.rfc-editor.org/rfc/rfc5452) – "Measures for Making DNS More Resilient against Forged Answers," *Proposed Standard*.
- RFC 5890 (https://www.rfc-editor.org/rfc/rfc5890) – "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework," *Proposed Standard*.
- RFC 5891 (https://www.rfc-editor.org/rfc/rfc5891) – "Internationalized Domain Names in Applications (IDNA): Protocol," *Proposed Standard*.
- RFC 5892 (https://www.rfc-editor.org/rfc/rfc5892) – "The Unicode Code Points and Internationalized Domain Names for Applications (IDNA)," *Proposed Standard*.
- RFC 5893 (https://www.rfc-editor.org/rfc/rfc5893) – "Right-to-Left Scripts for Internationalized Domain Names for Applications (IDNA)," *Proposed Standard*.

- RFC 6672 (https://www.rfc-editor.org/rfc/rfc6672) – "DNAME Redirection in the DNS," *Proposed Standard*.
- RFC 6891 (https://www.rfc-editor.org/rfc/rfc6891) – "Extension Mechanisms for DNS (EDNS(0))," *Internet Standard 75*.
- RFC 7766 (https://www.rfc-editor.org/rfc/rfc7766) – "DNS Transport over TCP - Implementation Requirements," *Proposed Standard*.
- RFC 8490 (https://www.rfc-editor.org/rfc/rfc8490) – "DNS Stateful Operations," *Proposed Standard*.
- RFC 8945 (https://www.rfc-editor.org/rfc/rfc8945) – "Secret Key Transaction Authentication for DNS (TSIG)," *Internet Standard 93*.
- RFC 9103 (https://www.rfc-editor.org/rfc/rfc9103) – "DNS Zone Transfer over TLS," *Proposed Standard*.
- RFC 9156 (https://www.rfc-editor.org/rfc/rfc9156) – "DNS Query Name Minimisation to Improve Privacy," *Proposed Standard*.

Proposed security standards

- RFC 4033 (https://www.rfc-editor.org/rfc/rfc4033) – "DNS Security Introduction and Requirements," *Proposed Standard*.
- RFC 4034 (https://www.rfc-editor.org/rfc/rfc4034) – "Resource Records for the DNS Security Extensions," *Proposed Standard*.
- RFC 4035 (https://www.rfc-editor.org/rfc/rfc4035) – "Protocol Modifications for the DNS Security Extensions," *Proposed Standard*.
- RFC 4470 (https://www.rfc-editor.org/rfc/rfc4470) – "Minimally Covering NSEC Records and DNSSEC On-line Signing," *Proposed Standard*.
- RFC 4509 (https://www.rfc-editor.org/rfc/rfc4509) – "Use of SHA-256 in DNSSEC Delegation Signer (DS) Resource Records (RRs)," *Proposed Standard*.
- RFC 5155 (https://www.rfc-editor.org/rfc/rfc5155) – "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence," *Proposed Standard*.
- RFC 5702 (https://www.rfc-editor.org/rfc/rfc5702) – "Use of SHA-2 Algorithms with RSA in DNSKEY and RRSIG Resource Records for DNSSEC," *Proposed Standard*.
- RFC 5910 (https://www.rfc-editor.org/rfc/rfc5910) – "Domain Name System (DNS) Security Extensions Mapping for the Extensible Provisioning Protocol (EPP)," *Proposed Standard*.
- RFC 5933 (https://www.rfc-editor.org/rfc/rfc5933) – "Use of GOST Signature Algorithms in DNSKEY and RRSIG Resource Records for DNSSEC," *Historic*. Changed to Historic status in

2024 by RFC 9558 (https://www.rfc-editor.org/rfc/rfc9558) . Updated by RFC 6944 (https://www.rfc-editor.org/rfc/rfc6944) . (See under Informational RFCs)

- RFC 7830 (https://www.rfc-editor.org/rfc/rfc7830) – "The EDNS(0) Padding Option," *Proposed Standard*.
- RFC 7858 (https://www.rfc-editor.org/rfc/rfc7858) – "Specification for DNS over Transport Layer Security (TLS)," *Proposed Standard*.
- RFC 8310 (https://www.rfc-editor.org/rfc/rfc8310) – "Usage Profiles for DNS over TLS and DNS over DTLS," *Proposed Standard*.
- RFC 8484 (https://www.rfc-editor.org/rfc/rfc8484) – "DNS Queries over HTTPS (DoH)," *Proposed Standard*.

Experimental RFCs

- RFC 1183 (https://www.rfc-editor.org/rfc/rfc1183) – "New DNS RR Definitions," *Experimental*.

Best Current Practices

- RFC 2182 (https://www.rfc-editor.org/rfc/rfc2182) – "Selection and Operation of Secondary DNS Servers," *Best Current Practice 16*.
- RFC 2317 (https://www.rfc-editor.org/rfc/rfc2317) – "Classless IN-ADDR.ARPA delegation," *Best Current Practice 20*.
- RFC 5625 (https://www.rfc-editor.org/rfc/rfc5625) – "DNS Proxy Implementation Guidelines," *Best Current Practice 152*.
- RFC 6895 (https://www.rfc-editor.org/rfc/rfc6895) – "Domain Name System (DNS) IANA Considerations," *Best Current Practice 42*.
- RFC 7720 (https://www.rfc-editor.org/rfc/rfc7720) – "DNS Root Name Service Protocol and Deployment Requirements," *Best Current Practice 40*.
- RFC 9499 (https://www.rfc-editor.org/rfc/rfc9499) – "DNS Terminology," *Best Current Practice 219*.

Informational RFCs

These RFCs are advisory in nature, but may provide useful information despite defining neither a standard or BCP.[1]

- RFC 1178 (https://www.rfc-editor.org/rfc/rfc1178) – "Choosing a Name for Your Computer," *Informational, For Your Information 5*.
- RFC 1591 (https://www.rfc-editor.org/rfc/rfc1591) – "Domain Name System Structure and Delegation," *Informational*.

- RFC 1912 (https://www.rfc-editor.org/rfc/rfc1912) – "Common DNS Operational and Configuration Errors," *Informational*.
- RFC 2100 (https://www.rfc-editor.org/rfc/rfc2100) – "The Naming of Hosts," *Informational*.
- RFC 3696 (https://www.rfc-editor.org/rfc/rfc3696) – "Application Techniques for Checking and Transformation of Names," *Informational*.
- RFC 3833 (https://www.rfc-editor.org/rfc/rfc3833) – "Threat Analysis of the Domain Name System (DNS)," *Informational*.
- RFC 4892 (https://www.rfc-editor.org/rfc/rfc4892) – "RIPv2 Cryptographic Authentication," *Informational*.
- RFC 5894 (https://www.rfc-editor.org/rfc/rfc5894) – "Internationalized Domain Names for Applications (IDNA): Background, Explanation, and Rationale," *Informational*.
- RFC 5895 (https://www.rfc-editor.org/rfc/rfc5895) – "Mapping Characters for Internationalized Domain Names in Applications (IDNA) 2008," *Informational*.
- RFC 8806 (https://www.rfc-editor.org/rfc/rfc8806) – "Running a Root Server Local to a Resolver," *Informational*.
- RFC 9076 (https://www.rfc-editor.org/rfc/rfc9076) – "DNS Privacy Considerations," *Informational*.
- RFC 9558 (https://www.rfc-editor.org/rfc/rfc9558) – "Use of GOST 2012 Signature Algorithms in DNSKEY and RRSIG Resource Records for DNSSEC," *Informational*.

Unknown

These RFCs have an official status of Unknown, but due to their age are not clearly labeled as such.

- RFC 920 (https://www.rfc-editor.org/rfc/rfc920) – "Domain requirements," *Status Unknown*. – Specified original top-level domains
- RFC 1032 (https://www.rfc-editor.org/rfc/rfc1032) – "DOMAIN ADMINISTRATORS GUIDE," *Status Unknown*.
- RFC 1033 (https://www.rfc-editor.org/rfc/rfc1033) – "DOMAIN ADMINISTRATORS OPERATIONS GUIDE," *Status Unknown*.
- RFC 1101 (https://www.rfc-editor.org/rfc/rfc1101) – "DNS Encoding of Network Names and Other Types," *Status Unknown*.

1. C. Huitema; J. Postel; S. Crocker (April 1995). *Not All RFCs are Standards* (https://datatracker.ietf.org/doc/html/rfc1796) . Network Working Group. doi:10.17487/RFC1796 (https://doi.

org/10.17487%2FRFC1796) . RFC 1796 (https://datatracker.ietf.org/doc/html/rfc1796) . *Informational*.

External links

- Vixie, Paul (4 May 2007). "DNS Complexity" (https://queue.acm.org/detail.cfm?id=1242499) . ACM Queue. Archived (https://web.archive.org/web/20230329160719/https://queue.acm.org/detail.cfm?id=1242499) from the original on 29 March 2023.
- Ball, James (28 February 2014). "Meet the seven people who hold the keys to worldwide internet security" (https://www.theguardian.com/technology/2014/feb/28/seven-people-keys-worldwide-internet-security-web) . *The Guardian*. Guardian News & Media Limited. Retrieved 28 February 2014.
- Kruger, Lennard G. (18 November 2016). "Internet Governance and the Domain Name System: Issues for Congress" (https://sgp.fas.org/crs/misc/R42351.pdf) (PDF). *Congressional Research Service*. Retrieved 27 July 2024.
- Zytrax.com (http://www.zytrax.com/books/dns/) , Open Source Guide – DNS for Rocket Scientists.
- Mess with DNS (https://messwithdns.net/) – site where you can do experiments with DNS.